SinkrOn
JURNAL & PENELITIAN TEKNIK INFORMATIKA
Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.15008
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Comparative Performance of YOLOv8 and SSD-mobilenet Algorithms for Road Damage Detection in Mobile Applications

Arie Satia Dharma[1)*], **Chantika Nadya Serebella Pardosi**[2)], **Zan Peter Silaen**[3)]
[1)2)3)] Institut Teknologi Del, Laguboti, Indonesia
[1)]ariesatia@del.ac.id, [2)]chantikanadya3@email.com, [3)] zan.silaen@del.ac.id

Submitted :July 8, 2025 | **Accepted** : July 27, 2025 | **Published** : July 28, 2025

Abstract: Road damage is a serious issue that can impede traffic and increase the risk of accidents in any area. Fast and accurate detection and classification of road damage are crucial for efficient maintenance and repair. Considering the ease of access, the implementation of this detection can be done using a mobile application. This study aims to compare the performance of two object detection algorithms, YOLOv8 and SSD-MobileNet, in detecting and classifying road damage in mobile application. Evaluation is conducted using accuracy, speed, and memory utilization, and classification of road damage into six categories namely block cracks, alligator cracks, transverse cracks, edge cracks, patches, and potholes using a confusion matrix. The results show that YOLOv8 has overall accuracy 86.4%, speed 0.5 ms, and consumes 0.41 GB of RAM. SSD-MobileNet shows an overall accuracy of 91.1%, speed 0.7 ms, and consumes 0.14 GB of RAM. The comparison indicates that YOLOv8 excels in detection speed, while SSD-MobileNet is more higher accuracy and efficient in memory. This study is limited to a performance measurement approach for YOLOv8 and SSD-MobileNet algorithms in a mobile-based road defect detection context. Its contribution lies in the trade-off between accuracy, speed, and the memory required to implement the models in limited devices. In future research is recommended to explore model with pruning to reduce memory usage.

Keywords: YOLOv8; SSD-MobileNet; object detection; classification; road damage.

INTRODUCTION

Road defects pose a significant problem, impeding traffic flow and elevating accident risks across various regions. Rapid and accurate detection and classification of these defects are crucial for efficient maintenance and repair. Damaged road infrastructure not only inconveniences road users but can also negatively impact a region's economy through increased repair costs and decreased productivity due to travel delays (Toyip Setiawan & Winayati, 2021). Therefore, identifying solutions for quick and accurate road defect identification is vital to minimize these adverse effects.

In recent years, the advancements in Artificial Intelligence (AI)-based technology enables automatic detection and classification of road defects by leveraging machine learning algorithms to analyze road images and identify defect types. Various algorithms can be employed for object detection in images, including RCNN, Faster RCNN, YOLO, and SSD. These algorithms are generally categorized into two types: one-stage and two-stage detectors (Carranza-García et al., 2021). One-stage detectors, such as YOLO and SSD, are designed for speed and efficiency, while two-stage detectors, like RCNN and Faster RCNN, prioritize detection accuracy, albeit at a slower speed (Sirisha et al., 2023).

Several previous studies have explored the application of AI technology for road defect detection. For instance, Zhang et al. (2018) utilized YOLOv3 for road defect detection and reported promising results in terms of detection speed. However, this study focused solely on one algorithm without comparing it to others. Another study by Liu et al. (2019) employed SSDs for road defect detection and found that SSDs could detect defects with good accuracy, but it did not address memory usage efficiency on mobile devices. In a more recent study, Chen et al. (2020) compared several object detection algorithms for road defect detection applications but did not include the latest versions of YOLO or SSD-MobileNet in their comparison.

*Arie Satia Dharma

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.15008
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Despite extensive research in this area, a significant gap remains in the literature regarding a direct and comprehensive comparison between the YOLOv8 and SSD-MobileNet algorithms specifically within the context of mobile applications for road defect detection. The mobile applications have limitations in terms of memory, computing power, and energy requirements. Previous studies have often focused on single algorithms or older versions, overlooking the advancements and unique challenges posed by mobile deployment, particularly concerning resource efficiency. For example, while some research has highlighted YOLO's speed (Zhang et al., 2018) and SSD's accuracy (Liu et al., 2019), a direct comparative analysis that considers the latest iterations of these models alongside their resource footprint on mobile devices is lacking. Furthermore, existing comparisons often omit the crucial aspect of memory usage, which is a critical constraint for mobile implementations (Chen et al., 2020).

This study aims to address this critical gap by conducting a comprehensive analysis of the performance of both YOLOv8 and SSD-MobileNet. The novelty of this research lies in its direct comparative evaluation of these two state-of-the-art object detection algorithms specifically optimized for mobile environments, considering not only detection accuracy and speed but also crucial factors like RAM usage and application storage size. This research provides a unique and timely contribution by offering practical insights into the trade-offs between speed, accuracy, and resource efficiency for road defect detection on mobile platforms, thereby filling a vital void in the current literature. This research will make an important contribution to the field of AI technology for road defect detection and is expected to provide better solutions for more efficient and effective road infrastructure maintenance.

LITERATURE REVIEW

YOLO (You Only Look Once) and SSD (Single Shot MultiBox Detector), which perform detection and classification in a single forward pass, offering superior speed and computational efficiency, though sometimes with a trade-off in accuracy, especially for small or overlapping objects (Carranza-García et al., 2021; Sirisha et al., 2023).

Prior studies have demonstrated the applicability of YOLO and SSD for road defect detection, highlighting their respective strengths. Zhang et al. (2018), for instance, utilized YOLOv3 to effectively detect various road defects, emphasizing its impressive speed for real-time applications. However, this work primarily focused on speed metrics without a detailed analysis of accuracy trade-offs or the feasibility of deployment on resource-limited mobile devices. Similarly, Liu et al. (2019) applied SSD for detecting cracks and potholes, reporting good accuracy. Yet, their research did not extensively evaluate the computational efficiency or memory footprint, which are paramount considerations for mobile-based solutions. While broader comparisons, such as Chen et al. (2020), have explored the performance of Faster RCNN, YOLOv3, and SSD, they often did not include the very latest iterations of these algorithms or specialized lightweight variants, leaving a gap in understanding their performance in modern, mobile-centric contexts.

This collective body of work underscores an inherent tension: achieving high detection accuracy often comes at the cost of computational efficiency, and vice-versa. While two-stage detectors generally offer superior accuracy, their computational demands often render them impractical for real-time mobile deployment. Conversely, one-stage detectors are more suitable for mobile devices due to their speed, but historical versions might exhibit reduced accuracy, particularly for subtle or fragmented road damage.

In response to these challenges, continuous advancements have led to newer models designed to better balance these trade-offs, making them more viable for real-world mobile applications. YOLOv8, a recent iteration released in 2023, incorporates significant architectural improvements aimed at enhancing both accuracy and efficiency (Lou et al., 2023; Drantantiyas et al., 2023). These advancements include a mosaic augmentation technique employed during training to increase data diversity and improve generalization, an innovative anchor-free detection head that simplifies the model's structure and boosts its performance, and an improved backbone network for more robust feature extraction. Concurrently, SSD-MobileNet integrates the efficient Single Shot MultiBox Detector (SSD) framework with the lightweight MobileNet backbone. MobileNet's architecture, characterized by its depthwise separable convolutions, significantly reduces the number of parameters and computational operations, thereby curtailing memory and computational demands. This design makes SSD-MobileNet particularly well-suited for deployment on resource-constrained mobile devices where efficiency is paramount (Falah et al., 2021; Pakpahan & Dewi, 2021). Despite the promising potential of both YOLOv8 and SSD-MobileNet for road defect detection, there has been limited direct comparative research evaluating these two specific, state-of-the-art models within the crucial context of mobile application deployment.

The existing literature therefore highlights several important gaps. Previous studies have often concentrated on individual algorithms rather than conducting comprehensive comparative analyses, and there is a notable scarcity of exploration into newer models like YOLOv8 and SSD-MobileNet, especially when considering the stringent memory and processing limitations inherent to mobile environments. Furthermore, most research has tended to emphasize accuracy or speed in isolation, failing to holistically evaluate all three essential aspects:

*Arie Satia Dharma

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.15008
e-ISSN : 2541-2019
p-ISSN : 2541-044X

detection accuracy, processing speed, and memory usage efficiency, which are all critical for practical mobile deployment.

This study directly addresses these identified gaps by undertaking a comprehensive comparative analysis of YOLOv8 and SSD-MobileNet specifically tailored for real-time road defect detection in mobile applications. The research will rigorously evaluate both models across all crucial performance metrics: detection accuracy for various types of road defects, detection speed to confirm suitability for real-time operational use, and memory and computational efficiency to ensure practical deployment on mobile devices. By integrating insights from prior research and diligently tackling these identified deficiencies, this study aims to provide clear guidance for practitioners and researchers in selecting the most appropriate AI model for the development of efficient, accurate, and practical mobile-based road defect detection systems. Recognizing the shortcomings of existing literature, this study aims to fill these gaps by providing a thorough comparative analysis of advanced, mobile-friendly object detection models, ensuring a holistic evaluation of accuracy, speed, and memory usage, which is crucial for real-world mobile deployment of road defect detection systems.

METHOD

In this research, methodology section discusses in detail start from the dataset collecting. After obtaining the image data, the next step is to extend it through augmentation and design the YOLOv8 and SSD-MobileNet algorithm models. Next, the models were tested against the classified road defect dataset. Evaluation is done by utilizing precision, recall, and mean average precision (mAP) metrics to evaluate the approach used. The research flow is presented in Figure 1.

Fig. 1 Research design

Data Collection

Sub In this research, a dataset of road defect images consisting of six types of defects is required: box cracks, crocodile skin cracks, transverse cracks, edge cracks, patches, and potholes. The data was collected from the free online platform roboflow, which provides high quality datasets. Each type of damage has 300 images, so the total dataset consists of 1800 images.

Preprocessing Data

Once the data is collected, the next step is image preprocessing. At this stage, data augmentation is performed to increase the variety of training data and prevent overfitting. The augmentation techniques used include rotation, cropping, noise addition, lighting changes, and horizontal flip. This process aims to make the model more robust to various image conditions that may be encountered in the field.

The processed images are then normalized to ensure consistency of input into the model. Normalization is done by rescaling the image pixel values so that they are within a certain range, such as 0 to 1. This process is important to improve the model's performance in detecting road damage.

The dataset was then divided into two main parts: training data and testing data. This is done in a ratio of 80:20, where 80% of the data is used to train the model and the remaining 20% is used to test the model's performance. This division aims to ensure that the model can be accurately evaluated based on data that has never been seen before.

Algorithm Implementation

After the data collection and processing process is complete, the next step is the implementation of the YOLOv8 and SSD-MobileNet algorithms. These two algorithms were chosen because they have the advantage of detecting objects in real-time and are efficient in memory usage.

*Arie Satia Dharma

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.15008
e-ISSN : 2541-2019
p-ISSN : 2541-044X

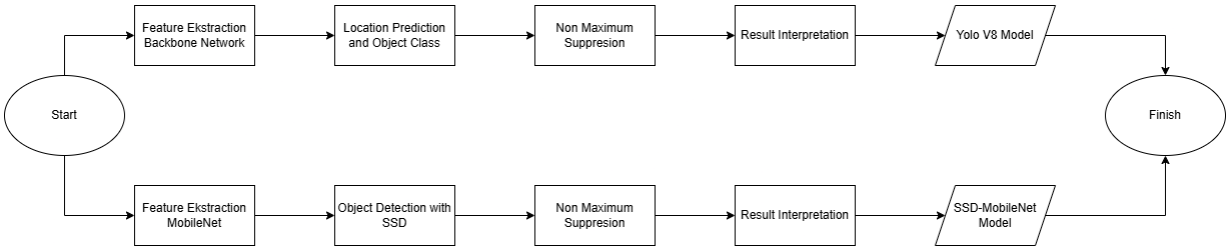

Fig. 2 YOLOv8 and SSD-MobileNet Algorithm Implementation Design

The object detection process using YOLOv8 involves several structured stages. First, image pre-processing is performed, including normalization and resizing, to prepare the input for the model. Then, utilizing a backbone network such as Darknet-53, important features are extracted from the images. Prediction of object location and class is then performed using convolution layers, followed by the application of non-max suppression techniques to reduce overlap and select the most relevant bounding box. After that, the post-processing step involves adding class labels and bounding box generation to the detected objects. The detection results are then interpreted and the performance of the model is evaluated by comparing the detection results with the ground truth data using relevant evaluation metrics such as precision and recall. Through this series of well-coordinated stages, YOLOv8 is able to provide object detection with high accuracy and optimal inference speed, making it one of the popular choices in image processing and computer vision.

SSD-MobileNet uses a structured approach in detecting objects in images. The first stage involves feature extraction using MobileNet, which is a lightweight and efficient convolutional neural network architecture. With MobileNet as the backbone, the model is able to retrieve features at various scales and resolutions through appropriate convolution layers. Furthermore, using SSD as its object detection method, SSD-MobileNet simultaneously predicts the bounding box and class score for each object in the image. This detection process is followed by the application of the Non-maximum Suppression (NMS) algorithm to overcome duplicate detection and select the most relevant bounding box. Finally, the model performs object classification by associating each bounding box with a class label based on the resulting class score. Through this series of steps, SSD-MobileNet is able to quickly and efficiently detect objects in images with good accuracy.

Table 1. Configuration Model of Yolov8

Parameter Configuration	Description
Optimizer	SGD (Stochastic Gradient Descent)
Learning Rate (Initial)	0.01
Epochs	100
Batch Size	8, 16, 32, 64 (Varied in hyperparameter tuning experiments)
Loss Function	Combined loss

Table 1 outlines the specific parameters utilized for training the YOLOv8 model in this study. The model employed was the YOLOv8 architecture, specifically initiating from a pre-trained yolov8n.pt base, a choice often made for its balanced performance, prioritizing both detection speed and accuracy for various applications (Lou et al., 2023). Training was conducted using the SGD optimizer, with an initial learning rate of 0.01 and a cosine annealing schedule, a common strategy to optimize convergence and prevent overfitting. The model was trained for 100 epochs with a batch size of 16, and both these parameters were subject to further tuning to achieve optimal performance, with the training performed on a Quadro P4000 GPU.

Table 2. Configuration Model of SSD-MobileNet

Parameter Configuration	Description
Optimizer	Adam
Learning Rate (Initial)	1
Epochs	100

*Arie Satia Dharma

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.15008
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Batch Size	8, 16, 32, 64 (Varied in hyperparameter tuning experiments)
Loss Function	MultiBox Loss
Num_Steps	20,000

Table 2 details the specific configuration used for the SSD-MobileNet model. This model integrates the Single Shot MultiBox Detector (SSD) head with a MobileNetV2 backbone, which was pre-trained on the COCO dataset. This combination is particularly favored for its lightweight design and efficiency, making it highly suitable for deployment on resource-constrained mobile devices (Falah et al., 2021; Pakpahan & Dewi, 2021). The model was trained using the Adam optimizer with an initial learning rate of 0.001, employing a step decay strategy for learning rate adjustment.

The training process for SSD-MobileNet involved extensive data augmentation techniques, including random cropping, horizontal flipping, and various photometric distortions to ensure the model's adaptability to varied image conditions. Training spanned 100 epochs with a batch size of 32, with the Num_Steps parameter (common in TensorFlow-based training) also being a focus for hyperparameter tuning. The efficient design and specific training parameters highlight its suitability for scenarios where memory and computational power are primary considerations.

Evaluation Method

The evaluation begins with the calculation of precision, recall, and mean average precision (mAP) for both YOLOv8 and SSD-MobileNet algorithms. Precision measures the proportion of true positive detections among all positive detections made by the model. It indicates how many of the detected road damages are actual damages. Recall measures the proportion of true positive detections among all actual road damages in the dataset, indicating the model's ability to detect all possible damages. Mean Average Precision (mAP) is used to summarize the precision-recall curve and is calculated by taking the average precision across various recall levels. The mAP@0.5 metric is specifically used in this study, which considers a detection correct if the Intersection over Union (IoU) between the predicted bounding box and the ground truth is greater than 0.5.

The dataset is split into training and testing sets with a ratio of 80:20. The training set, comprising 80% of the dataset, is used to train the models, while the remaining 20% is reserved for testing and evaluation. Ground truth labels, which provide the actual locations and classes of road damages, are used as a benchmark to compare against the predictions made by the models. The evaluation metrics are computed by comparing these ground truth labels with the model predictions on the test set.

The inference speed of the models is a critical factor, especially for real-time applications. Inference speed is measured as the time taken to process one image and is expressed in milliseconds (ms) per frame. This metric is crucial for applications where quick decision-making is necessary, such as real-time road damage detection on mobile devices.

Memory usage is evaluated in terms of both storage and RAM consumption. The storage usage is measured before and after converting the models to TensorFlow Lite format, which is necessary for deployment on mobile devices. The TensorFlow Lite conversion process often reduces the model size significantly, making it more suitable for mobile deployment. RAM usage is measured to determine the amount of memory required during the inference process, which impacts the performance of mobile devices with limited resources.

The final step in the evaluation involves a comparative analysis of both algorithms based on the collected metrics. The accuracy, speed, and memory usage efficiency of YOLOv8 and SSD-MobileNet are compared to determine which algorithm offers the best balance of performance.

RESULT

Road Damage Detection Results with YOLOv8 Algorithm

The YOLOv8 object detection model is conducted by using hyperparameters including batch size. The following table is the experimental results in forming a model for detecting and classifying road damage.

Table 3 Experiment Results

Model	Set Epoch	Batch_Size	Precision	Recall	mAP@.5
Yolov8	100	8	0.821	0.781	0.732
	100	16	0.851	0.728	0.789
	100	32	0.878	0.731	0.793
	100	**64**	**0.907**	**0.821**	**0.864**

*Arie Satia Dharma

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.15008
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Table 3 presents the performance of the YOLOv8 model across different batch sizes, while keeping the number of epochs constant at 100. The key metrics evaluated are Precision, Recall, and mean Average Precision at an Intersection over Union (IoU) threshold of 0.5 (mAP@.5). This experiment aims to illustrate how batch size influences the model's ability to accurately detect and classify objects.

As the batch size increases from 8 to 64, a clear trend of improving performance is observed. Precision consistently rises from 0.821 to 0.907, indicating that a larger batch size leads to fewer false positives. Similarly, mAP@.5, a comprehensive measure of object detection accuracy, shows a significant increase from 0.732 to 0.864. This suggests that with more data processed in each training iteration, the model learns more robust features, leading to better overall detection quality.

Interestingly, Recall also demonstrates an improvement, albeit with a slight dip at batch size 16, ultimately increasing from 0.781 to 0.821 at batch size 64. The most optimal performance, balancing all metrics, appears to be achieved at a batch size of 64, where YOLOv8 yielded the highest Precision (0.907), Recall (0.821), and mAP@.5 (0.864).

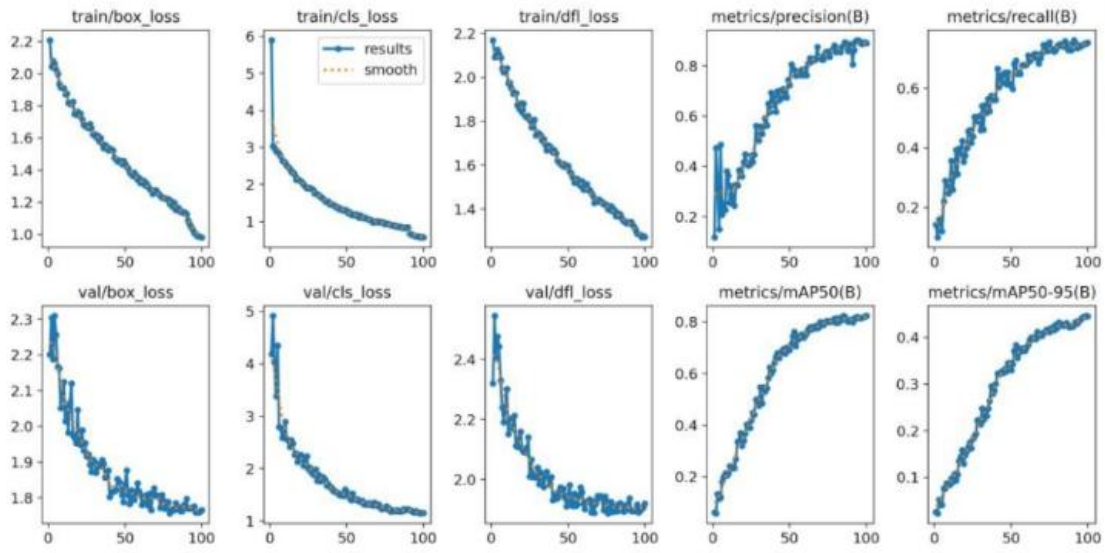

Fig. 3 YOLOv8 Algorithm Training Dataset Evaluation Results

Figure 3 shows the training data, shows that the model experienced a significant performance improvement over 100 training epochs. The train/box_loss, train/cls_loss, and train/dfl_loss graphs show a consistent decrease in loss values from the beginning to the end of training. This decrease indicates that the model is improving at determining object bounding boxes, classifying objects, and capturing local object features over time.

Fig. 4 Detection Result of YOLOv8 Algorithm

The Figure 4 shows the results of road damage detection by the YOLOv8 algorithm, displaying various damage types such as Side Cracks, Potholes, Crocodile Skin Cracks, Longitudinal Cracks, and Patches with

*Arie Satia Dharma

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.15008
e-ISSN : 2541-2019
p-ISSN : 2541-044X

colored bounding boxes and corresponding labels. YOLOv8 successfully detects defects in various lighting conditions and viewing angles, e.g. red box for potholes and yellow box for elongated cracks. These detections indicate YOLOv8's ability to identify road defects with high accuracy, showing great potential for road infrastructure monitoring and maintenance.

Road Damage Detection with SSD-MobileNet Algorithm

The SSD-MobileNet model also showed good evaluation results with precision between 0.736 to 0.916, recall between 0.768 to 0.942, and mAP@0.5 between 0.717 to 0.914. The total accuracy achieved was 91.1%. These results show that the SSD-MobileNet model has a higher performance in detecting road defects than YOLOv8. The average inference time for SSD-MobileNet is 60 ms per image, which also enables real-time detection.

Table 4 Experiment Results with Changing Num_Steps (Batch-Size=16)

Num_Steps	Precision	Recall	mAP@.5
10000	0.736	0.768	0.717
30000	0.864	0.898	0.816
50000	0.916	0.942	0.914

Table 4 presents the performance metrics for a model (likely SSD-MobileNet, given the parameter "Num_Steps") when varying the number of training steps while keeping the batch size constant at 16. The metrics evaluated are Precision, Recall, and mAP@.5. This experiment aims to illustrate the impact of the total training iterations on the model's accuracy. The results clearly indicate that increasing the number of training steps significantly improves the model's performance across all metrics. As "Num_Steps" increases from 10,000 to 50,000, Precision rises from 0.736 to 0.916, Recall improves from 0.768 to 0.942, and mAP@.5 sees a substantial increase from 0.717 to 0.914. This strong positive correlation demonstrates that longer training, up to 50,000 steps, allows the model to learn more complex patterns and features from the dataset, leading to superior object detection capabilities.

Table 5 Experiment Results with Changing Num_Steps (Num_Steps=20000)

Batch-Size	Precision	Recall	mAP@.5
8	0.857	0.874	0.742
16	0.864	0.898	0.816
32	**0.871**	**0.915**	**0.864**
64	0.891	0.908	0.851

Table 5 show that increasing the Batch-Size from 8 to 64 provides an improvement in Precision and Recall values, but the best mAP@.5 value is achieved at Batch-Size 32. This suggests that there is an optimal point where increasing the Batch-Size no longer provides a significant improvement in model performance.

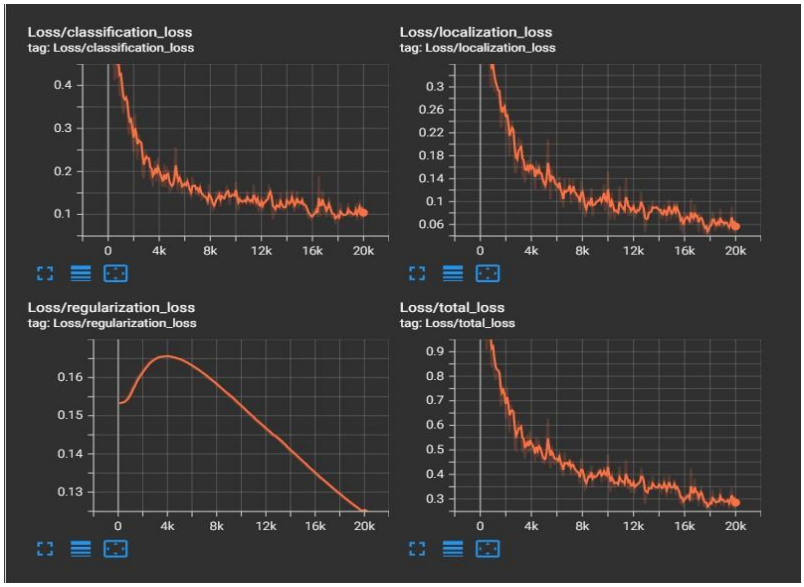

Fig. 5 SSD-MobileNet Training Loss Graph

*Arie Satia Dharma

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.15008
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Figure. 5 show the loss graphs that occurred during training of the SSD-MobileNet model. The loss graph illustrates the model's performance during training. The loss graph measures how well or poorly the model's predictions compare to the actual values. Classification loss and localization loss both show significant decreases, indicating improvements in the model's accuracy and precision. Regularization loss initially increased and then decreased, helping to avoid overfitting. Total loss, the combination of all three metrics, also decreased consistently, reflecting improvements in the model's overall performance. This graph indicates that the SSD-MobileNet model is improving at classifying and detecting objects while maintaining generalization, making it effective for road damage detection.

Fig. 6 Detection Results Using SSD-MobileNet Algorithm

Figure 6 shows the results of the pothole detection by SSD-MobileNet. The image shows the detection of potholes with 91% confidence. The SSD-MobileNet demonstrated good detection capabilities under varying surface and lighting conditions, providing clear visual information on the location and size of the defect. These results reflect the effectiveness of SSD-MobileNet in road maintenance, detecting defects with considerable precision, aiding early identification and efficient repair planning.

Comparison of Algoritma Performance

A performance comparison of the YOLOv8 and SSD-MobileNet algorithms is conducted based on the evaluation results which provide important insights into the performance of both models in key aspects such as accuracy, speed, and memory usage. YOLOv8 showed slightly lower accuracy with 86.4% compared to 91.1% of SSD-MobileNet having a higher accuracy rate, making it more effective in road defect detection. YOLOv8 excels in terms of detection speed with faster inference time, while SSD-MobileNet shows higher total accuracy and better memory usage efficiency. The implementation of YOLOv8 in the mobile application requires 136 MB of storage and 0.41 GB of RAM, while SSD-MobileNet requires 18.8 MB of storage and 0.14 GB of RAM. This shows that SSD-MobileNet is more suitable to be implemented on mobile devices with varying specifications.

Table 6 Comparison of the Two Algorithms from the Constructed Model

Indication	YOLOv8	SSD-MobileNet
Accuracy	86.4%	91.1%
Model Speed	0.5 ms	0.7 ms
Storage Usage (Mobile Application)	136 MB	18.8 MB
RAM Usage (Mobile App)	0.41 GB	0.14 GB

Based on Table 6 of this comparison, it is recommended to use SSD-MobileNet for the implementation of road defect detection in mobile applications due to its better efficiency. SSD-MobileNet demonstrates superior performance in terms of memory usage and storage requirements, making it an ideal choice for mobile devices with limited resources. The algorithm requires significantly less storage space and RAM, which is crucial for maintaining the performance and battery life of mobile devices. With SSD-MobileNet, the application can run smoothly without overburdening the device, ensuring a seamless user experience. Additionally, the high precision and recall rates achieved by SSD-MobileNet indicate that it can reliably detect various types of road defects, making it a robust solution for real-world scenarios.

*Arie Satia Dharma

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.15008
e-ISSN : 2541-2019
p-ISSN : 2541-044X

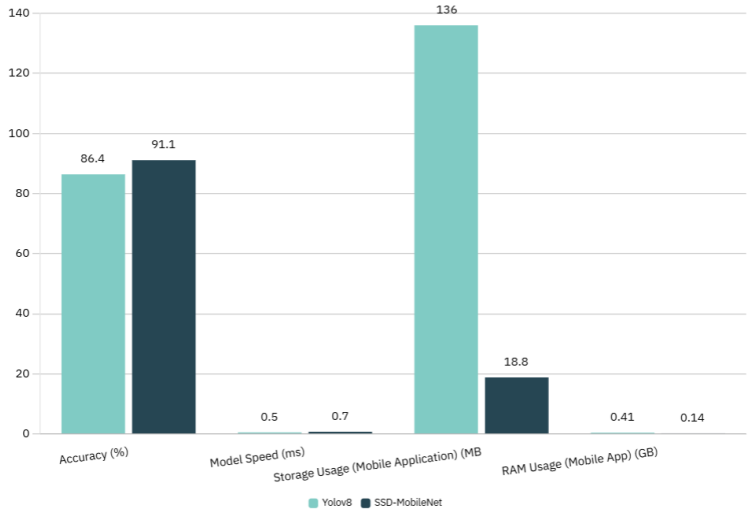

Fig. 7 Bar Chart Result Model

Figure 7 compares the performance of two object detection models, YOLOv8 and SSD-MobileNet, across four metrics: accuracy, model speed, storage usage, and RAM usage in a mobile application. SSD-MobileNet demonstrates higher accuracy (91.1%) than YOLOv8 (86.4%) but is slightly slower in model speed (0.7 ms vs. 0.5 ms). In terms of resource efficiency, SSD-MobileNet significantly outperforms YOLOv8, requiring much less storage (18.8 MB vs. 136 MB) and RAM (0.14 GB vs. 0.41 GB). Overall, SSD-MobileNet proves to be a more suitable choice for mobile applications due to its balance of high accuracy and efficient resource consumption.

DISCUSSIONS

This section discusses and interprets the presented results, compares them to previous research, and highlights key findings. We also provide technical justifications for the observed performance differences and suggest potential avenues for future research.

First, SSD-MobileNet demonstrated higher accuracy (91.1%) compared to YOLOv8 (86.4%). This confirms SSD-MobileNet's advantage in classifying diverse road defects, especially box cracking and transverse cracking. Its robust performance in defect classification makes it a strong candidate for applications where precise identification of various crack types is critical

Second, YOLOv8 had a faster detection speed (0.5 ms) than SSD-MobileNet (0.7 ms). This aligns with YOLO's architecture as a one-stage detector optimized for real-time performance. The superior speed of YOLOv8 can be attributed to its anchor-free and single-head prediction architecture. Unlike multi-stage detectors or those relying on anchor boxes, YOLOv8 directly predicts bounding boxes and class probabilities for each grid cell, streamlining the detection process and significantly reducing computational overhead. This design makes it highly suitable for applications requiring immediate feedback, such as live video analysis. Conversely, SSD-MobileNet, while highly accurate, uses anchor boxes, which contributes to its slightly longer processing time compared to YOLOv8's anchor-free approach.

Third, SSD-MobileNet significantly outperformed YOLOv8 in memory and storage efficiency, requiring less RAM (0.14 GB vs 0.41 GB) and a smaller application storage size (18.8 MB vs 136 MB). These findings are consistent with research highlighting MobileNet's lightweight design for mobile deployment. The remarkable efficiency of SSD-MobileNet is primarily due to the depthwise separable convolutions employed in its MobileNet backbone. This architectural innovation replaces standard convolutions with a two-step process: a depthwise convolution that applies a single filter per input channel, followed by a pointwise convolution (1×1) that combines the outputs of the depthwise convolution. This approach drastically reduces the number of parameters and computational operations, leading to a much smaller model footprint and lower memory consumption, making it ideal for resource-constrained mobile devices.

Moreover, experiments showed that increasing Epochs or Batch Size improved model performance to a point, but beyond certain thresholds, improvements were marginal. For SSD-MobileNet, Num_Steps had a large impact on precision and recall, showing the importance of careful hyperparameter tuning. These findings underscore the need for meticulous optimization during the training phase to maximize model effectiveness without incurring unnecessary computational costs.

*Arie Satia Dharma

e-ISSN : 2541-2019
p-ISSN : 2541-044X

Overall, the choice between these algorithms depends on specific application needs. For applications prioritizing resource efficiency and high accuracy on mobile devices, SSD-MobileNet is more suitable. For scenarios demanding real-time speed where resources are less constrained, YOLOv8 may be preferred.

Future Development

These results contribute to understanding how modern object detection models like YOLOv8 and SSD-MobileNet perform under real-world road damage detection scenarios, offering practical insights for developers and researchers aiming to implement AI in mobile road inspection tools. Looking ahead, potential future developments should focus on enhancing the algorithms themselves. This includes exploring advanced quantization techniques for both YOLOv8 and SSD-MobileNet to further reduce model size and improve inference speed on mobile devices without significant loss in accuracy. Additionally, investigating neural architecture search (NAS) methods could lead to the discovery of even more optimized and lightweight architectures specifically tailored for road defect detection on constrained mobile hardware. Future research could also involve developing hybrid models that combine the strengths of both one-stage (speed) and two-stage (accuracy) approaches, or integrating attention mechanisms into current architectures to improve their ability to focus on subtle defect features. Furthermore, exploring domain adaptation techniques would be crucial to ensure these models maintain high performance across diverse road conditions, lighting, and environmental factors, making them more robust for real-world deployment.

CONCLUSION

This study aims to compare the performance of two object detection algorithms, namely YOLOv8 and SSD-MobileNet, in detecting road defects. Based on the results obtained, both algorithms show good ability to detect various types of road defects with high accuracy. The YOLOv8 model shows an advantage in detection speed with an average inference time of 0.5 ms per frame, which enables real-time detection. However, there are limitations to YOLOv8, especially in terms of larger memory usage, which requires 136 MB of storage and 0.41 GB of RAM, making it more suitable for devices with a large enough memory capacity. On the other hand, SSD-MobileNet showed higher total accuracy with a mAP@0.5 value of 91.1%, compared to 86.4% for YOLOv8. In addition, SSD-MobileNet is more efficient in memory usage, requiring 18.8 MB of storage and 0.14 GB of RAM. Although the inference time of SSD-MobileNet is slightly slower at 0.7 ms per frame, the efficient use of memory and storage makes it more ideal for implementation on resource-constrained mobile devices. Although the results of this study are promising, there are some limitations that need to be considered. One of the main limitations is the detection capability of the model which tends to highlight the class with the highest confidence score. This causes difficulties in detecting more than two classes simultaneously. In addition, the process of labeling the datasets directly based on their respective classes can affect the detection results. For future research, it is recommended to improve the dataset labeling process in order to detect more than two classes more accurately. In addition, it is necessary to test on larger and more complex datasets to improve the accuracy and efficiency of both algorithms.

Thus, this research is expected to make a significant contribution to road infrastructure maintenance and repair efforts, as well as a reference for the development of road damage detection technology in the future. The results of this research also open up opportunities for further research in optimizing the use of object detection algorithms in various contexts and applications.

REFERENCES

Sirisha, U., Praveen, S. P., Srinivasu, P. N., Barsocchi, P., & Bhoi, A. K. (2023). Statistical analysis of design aspects of various YOLO-based deep learning models for object detection. International Journal of Computational Intelligence Systems, 16(1), 1–29. https://doi.org/10.1007/s44196-023-00302-w

Drantantiyas, N. D. G., Yulita, W., Ridwan, N. T., Ramadhani, U. A., Kesuma, R. I., Rakhman, A. Z., Bagaskara, R., Miranto, A., & Mufidah, Z. (2023). Performasi deteksi jumlah manusia menggunakan YOLOv8. JASIEK (Jurnal Aplikasi Sains, Informasi, Elektronika dan Komputer), 5(2), 63–68. https://doi.org/10.26905/jasiek.v5i2.11605

Carranza-García, M., Torres-Mateo, J., Lara-Benítez, P., & García-Gutiérrez, J. (2021). On the performance of one-stage and two-stage object detectors in autonomous vehicles using camera data. Remote Sensing, 13(1), 1–23. https://doi.org/10.3390/rs13010089

Falah, K., Gustiana, M., & Ungkawa, U. (2021). Karakteristik metode MobileNet-SSD dengan pre-trained model MobileNet untuk objek bergerak. Jurnal Nasional Institut Teknologi Nasional, 5(2), 45–55. https://doi.org/10.1234/journal.5678

*Arie Satia Dharma

Pakpahan, B., & Dewi, I. C. (2021). Pendeteksian lubang pada jalanan menggunakan metode SSD-MobileNet. Indonesian Journal of Electronics and Instrumentation Systems (IJEIS), 11(2), 213–222. https://doi.org/10.22146/ijeis.60157

Choudhari, V., Phadtare, M., & Karad, V. (2021). Comparison between YOLO and SSD MobileNet for object detection in a surveillance drone. International Journal of Scientific Research and Engineering Management, 5(10), 0–5. https://doi.org/10.13140/RG.2.2.34029.51688

Sabina, N. (2022). Object detection using YOLO and MobileNet SSD: A comparative study. International Journal, 11(6), 134–138.

Toyip Setiawan, F. S., & Winayati. (2021). Identifikasi jenis-jenis kerusakan jalan (perkerasan lentur). Jurnal, 6(1), 69–77.

Kim, J. A., Sung, J. Y., & Park, S. H. (2020). Comparison of Faster-RCNN, YOLO, and SSD for real-time vehicle type recognition. In 2020 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia) (pp. 8–11). IEEE. https://doi.org/10.1109/ICCE-Asia49877.2020.9277040

Nugroho, P. A., Fenriana, I., & Arijanto, R. (2020). Implementasi deep learning menggunakan convolutional neural network (CNN) pada ekspresi manusia. Algor, 2(1), 12–21.

Lou, H., et al. (2023). DC-YOLOv8: Small-size object detection algorithm based on camera sensor. Electronics, 12(10), 1–14. https://doi.org/10.3390/electronics12102323

Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, faster, stronger. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 6517–6525.

Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. arXiv. https://arxiv.org/abs/1804.02767

Wang, C., Bochkovskiy, A., & Liao, H. M. (2020). CSPNet: A new backbone that can enhance learning capability of CNN. arXiv. https://arxiv.org/abs/1911.11929

Iyer, R., Ringe, P. S., Iyer, R. V., & Bhensdadiya, K. P. (2021). Comparison of YOLOv3, YOLOv5s and MobileNet-SSD V2 for real-time mask detection. International Journal of Research in Engineering and Technology, 8(7), 1156–1160. https://www.researchgate.net/publication/353211011

Indra, D., Herman, H., & Budi, F. S. (2023). Implementasi sistem penghitung kendaraan otomatis berbasis computer vision. Komputika Jurnal Sistem Komputer, 12(1), 53–62. https://doi.org/10.34010/komputika.v12i1.9082

Kim, Y.-H., & Kim, Y. L. (2020). Comparison of CNN and YOLO for object detection. Journal of Semiconductor Display Technology, 19(1), 85–92.

Murthy, C. B., Hashmi, M. F., & Keskar, A. G. (2021). Optimized MobileNet + SSD: A real-time pedestrian detection on a low-end edge device. International Journal of Multimedia Information Retrieval, 10(3), 171–184. https://doi.org/10.1007/s13735-021-00212-7

Ulhaq, M. R. D., Zaidan, M. A., & Firdaus, D. (2023). Pengenalan ekspresi wajah secara real-time menggunakan metode SSD MobileNet berbasis Android. Jurnal Teknologi Informatika (JOTI), 5(1), 48–52. https://doi.org/10.37802/joti.v5i1.387

Krstinić, D., Braović, M., Šerić, L., & Božić-Štulić, D. (2020). Multi-label classifier performance evaluation with confusion matrix. Computer Science & Information Technology (CS & IT), 10(8), 1–14. https://doi.org/10.5121/csit.2020.100801

Sun, C., Zhang, H., Qin, S., Qin, J., Shi, Y., & Wen, Q. (2020). DroidPDF: The obfuscation resilient packer detection framework for Android apps. IEEE Access, 8, 167460–167474. https://doi.org/10.1109/ACCESS.2020.3022477

*Arie Satia Dharma